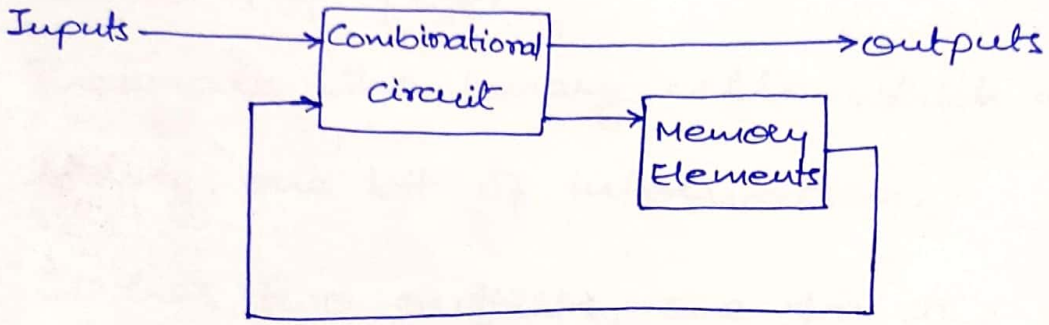
# SEQUENTIAL CIRCUITS

Block Diagram :→

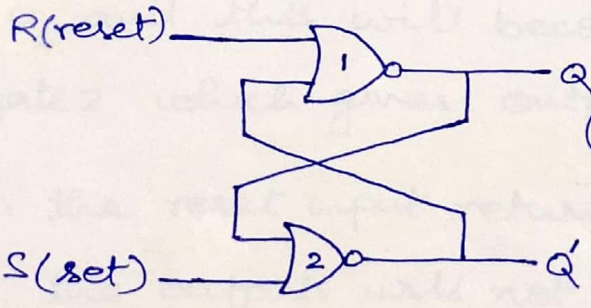



- Sequential circuits are defined as the circuits whose current output will not only depend on the current inputs but also on the previous output.
- Its block diagram consists of a Combinational circuit to which memory elements are connected to form a feedback path to provide the present outputs as a part of the next inputs.
- The memory elements are the devices which are capable of storing binary information within them.
- The binary information stored in the memory elements at any given time is defined as the state of the sequential circuit.
- The next state of the memory elements will be a function of external inputs and the present state.

## Flip-flop

- The memory elements used in sequential circuits are called flip-flops.
- These are the binary cells which are capable of storing one bit of information.
- It has two outputs, one for the normal value and one for the complemented value of the bit stored in it.
- Basic circuit of a flip-flop is constructed using either two NAND-Gates or two NOR-Gates.
- The two outputs of the flip-flop are designated as normal value, Q and complement value, Q'.
- Two inputs are Set (S) and Reset (R).

### Basic flip-flop circuit with NOR-Gates



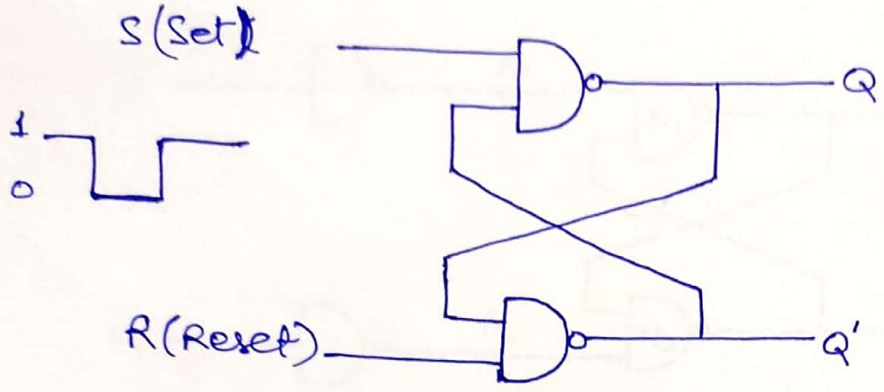

(a) Logic Diagram

| S | R | Q | Q' | |
|---|---|---|---|---|
| 1 | 0 | 1 | 0 | (Set state) |
| 0 | 0 | 1 | 0 | (after S=1, R=0) |
| 0 | 1 | 0 | 1 | |
| 0 | 0 | 0 | 1 | (after S=0, R=1) |
| 1 | 1 | 0 | 0 | undesired condition |

## Explanation for Truth-Table

- The output of a NOR gate is 0 if any input is 1 and the output is 1 only when all inputs are 0.
- Taking S = 1 and R = 0, we get output of NOR-gate 2 as 0 and this will become an input to NOR-gate 1 and which gives output as 1.
- After the previous state, we make S = 0, so we are now having S=0 and 1 as the second input to the NOR-gate 2. which gives output as 0. This will be the second input now to the NOR-gate 1 along with R=0. This will give 1 as output of NOR-gate 1.
  i.e. outputs do not change.
- Taking S=0 and R=1, we get output of NOR-gate 1 as 0 and this will become an input to NOR-gate 2 which gives output as 1.
- When the reset input returns to 0 after the previous state, the outputs will not change.
- When R=1 and S=1, both Q and Q' outputs go to 0 and since $Q \neq Q'$ so, this is undesired condition and it must be avoided.

# Basic flip-flop circuit with NAND-gates




(a.) Logic Diagram

| S | R | Q | Q' | |
|---|---|---|---|---|
| 1 | 0 | 0 | 1 | |
| 1 | 1 | 0 | 1 | (after S=1, R=0) |
| 0 | 1 | 1 | 0 | |
| 1 | 1 | 1 | 0 | (after S=0, R=1) |
| 0 | 0 | 1 | 1 | |

(b.) Truth Table

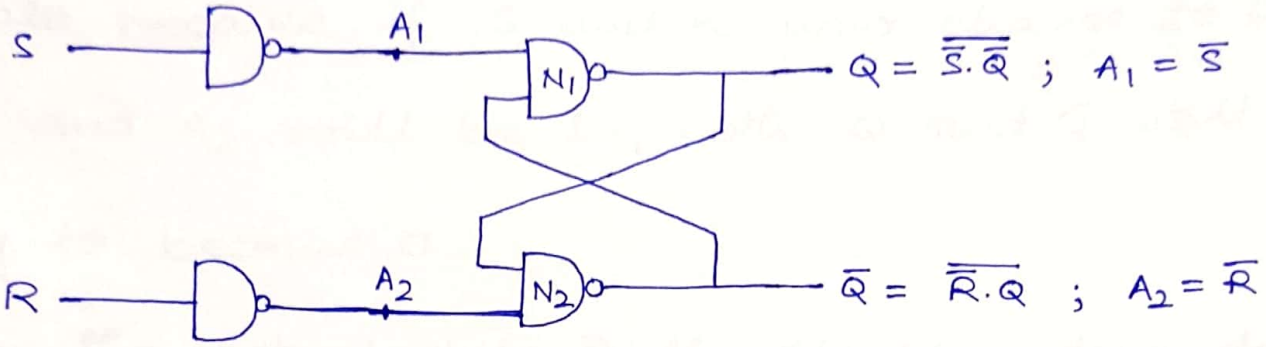
## Explanation for truth-table

- The output of a NAND-gate is 1 if any input is 0 and the output is 0 only when all inputs are 1.

- A flip-flop has two useful states. when Q=1 and Q'=0, it is in set-state (or 1-state).

when Q=0 and Q'=1, it is in the clear-state (or 0-state).

# States of a flip-flip




## SR flip-flop

(1.) **Set State**: when S = 1 and R = 0, $A_1$ becomes 0, making Q = 1. So, now $\bar{Q} = \overline{1.1} = 0$. This input condition sets the flip flop (Q = 1, $\bar{Q} = 0$); so it is called SET-STATE.

(2.) **Reset State**: when S = 0 and R = 1, $A_2$ becomes 0, making $\bar{Q} = 1$ and $Q = \overline{1.1} = 0$. This input condition resets the flip flop (Q = 0, $\bar{Q} = 1$); so it is called Reset-State.

(3.) **No change**: when S = R = 0; $Q = \overline{1.\bar{Q}} = Q$ and $\bar{Q} = 1.\bar{Q} = \bar{Q}$. The flip flop remains in whatever state it is in.

(4.) Race: When $S=R=1$, $A_1$ and $A_2$ both become 0. So $Q=\bar{Q}=1$. This is an undesired output state because if S and R now change to 0, $A_1$ and $A_2$ will be 1, both Q and $\bar{Q}$ will try to become 0.

Now the actual state of the flip flop depends on the relative delays of the two gates.

If $N_2$ is faster, $\bar{Q}$ will become $\overline{1.1}=0$ first and will make $Q=1$. and

If $N_1$ is faster, Q will become 0 first and will make $\bar{Q}=1$.

This is called as Race Condition. Here the state of the flip flop is uncertain, so this condition is not allowed.

## Latch V/S flip flop

## Lathes V/S Flipflop

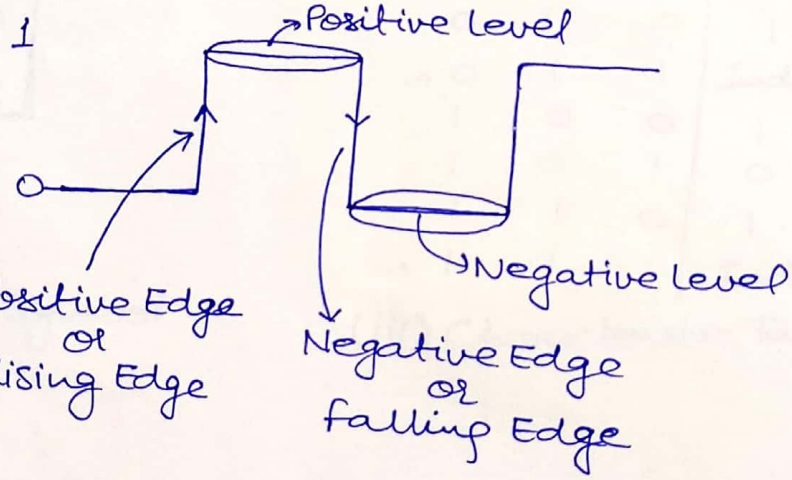
- Lathes are level-triggered while flip-flops are edge-triggered.
- Latches can have a clocked circuitary.
- Lathes that are clocked are known as Gated-Latch.

# Triggering of Flip-flops

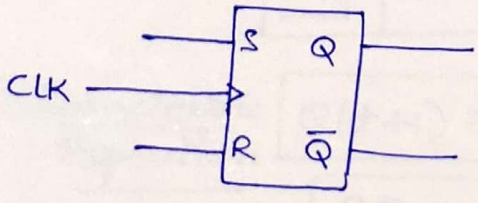
→ When the flip flop will give the output in a clocked circuit.

## Triggering Methods

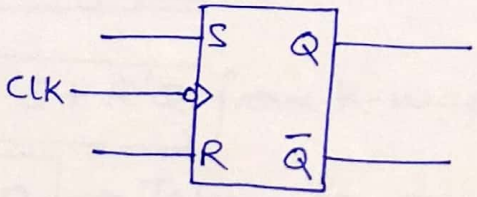
- Edge Triggering
  - Positive Edge Triggering
  - Negative Edge Triggering
- Level Triggering
  - Positive Level triggering
  - Negative Level Triggering




## Symbols



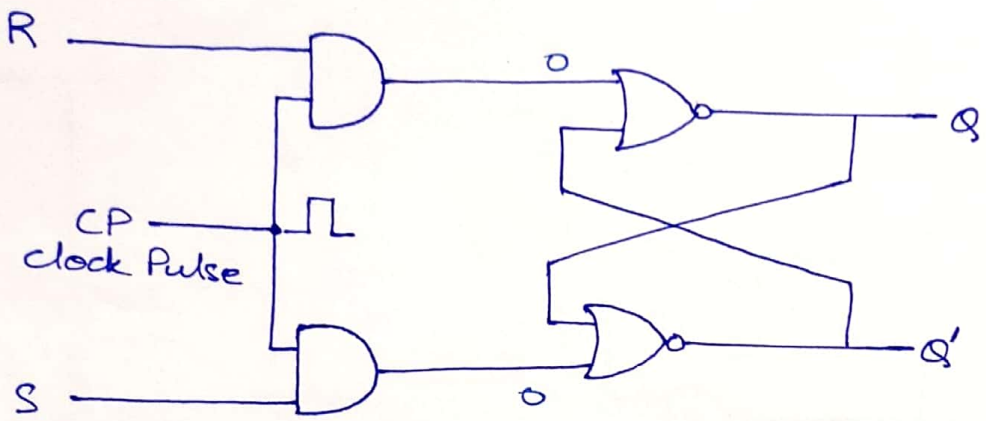

Positive-Edge Triggered SR-flip flop



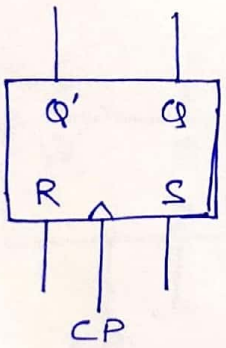

Negative-Edge Triggered SR-flip flop

Take 1 Example on waveform o/p of SR flip-flop

# Clocked RS Flip Flop




(i) Logic Diagram

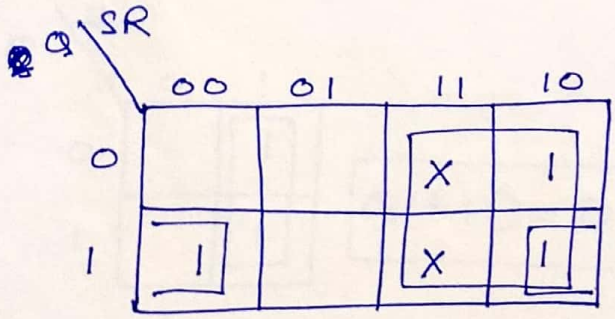



(ii) Graphic Symbol

$Q_n$ (Present state) → $Q_{n+1}$ (Next State)

| $Q(t)$ | S | R | $Q(t+1)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| → 0 | 1 | 1 | Indeterminate |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| → 1 | 1 | 1 | Indeterminate |

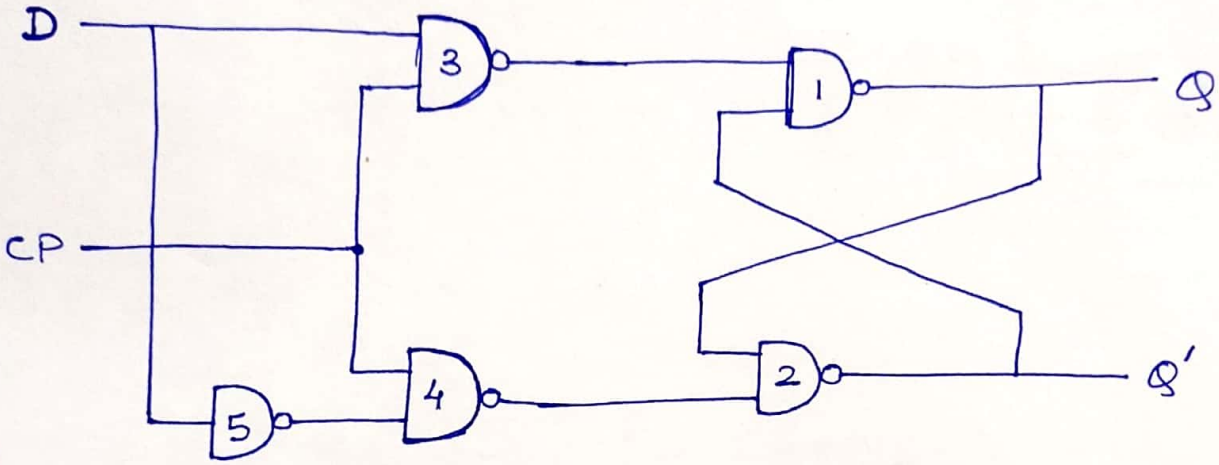
(iii) Characteristic Table



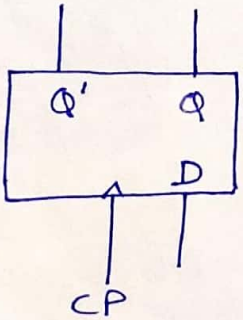

Characteristic Equation: $\boxed{Q(t+1) = S + R'Q}$ from K-map

$\boxed{SR = 0}$ → Taken to avoid the condition when $S = R = 1$

# D flip-flop



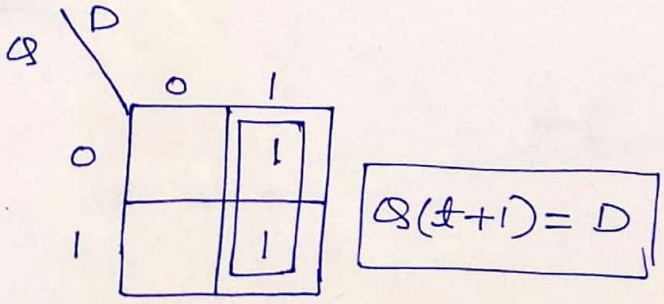

Logic Diagram




Graphic Symbol

| Q | D | Q(t+1) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Characteristic Table




$Q(t+1) = D$

Characteristic Equation

## Clocked SR Flip-flop

1 shows the clocked SR flip-flop. The circuit is similar to SR latch except enable signal is
d by clock pulse (CP). On the positive edge of the clock pulse. The circuit responds to the *S* and
.

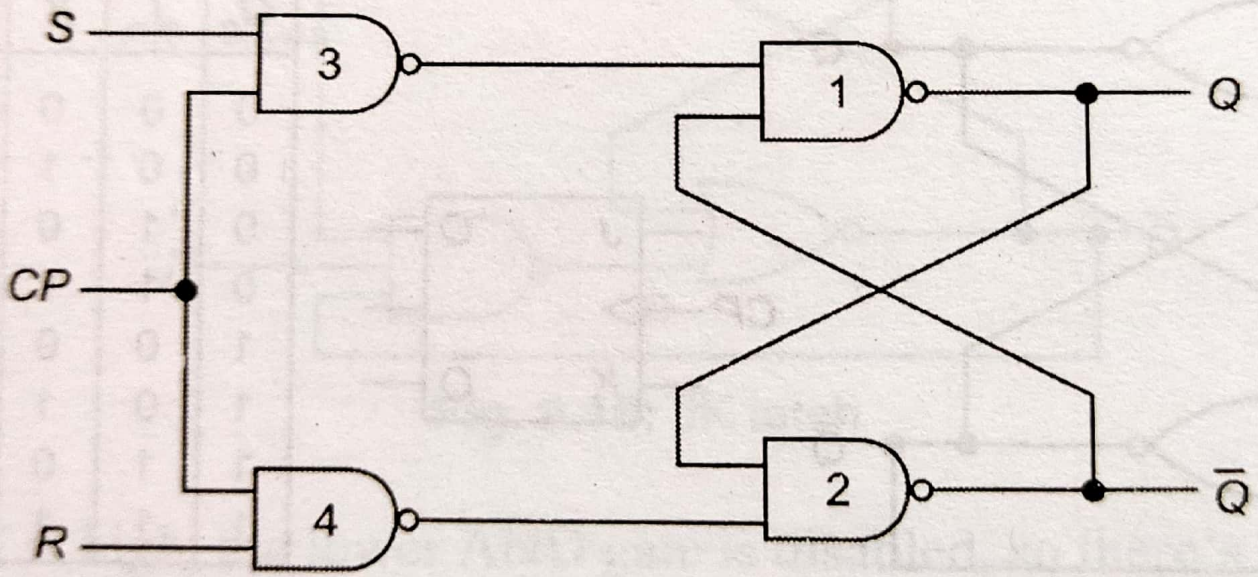
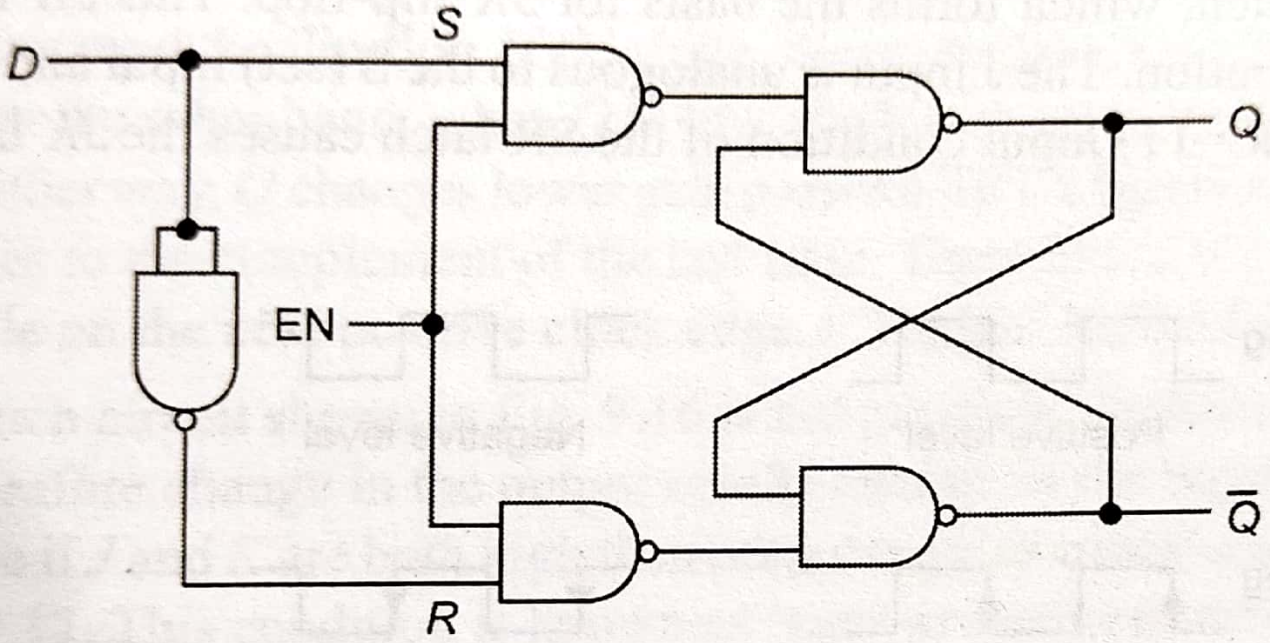
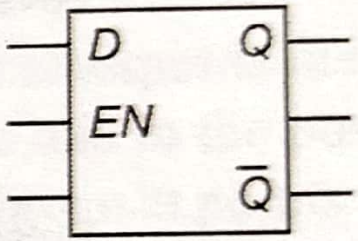



**Fig. 9.21:** Clocked SR flip-flop




(a) Latch

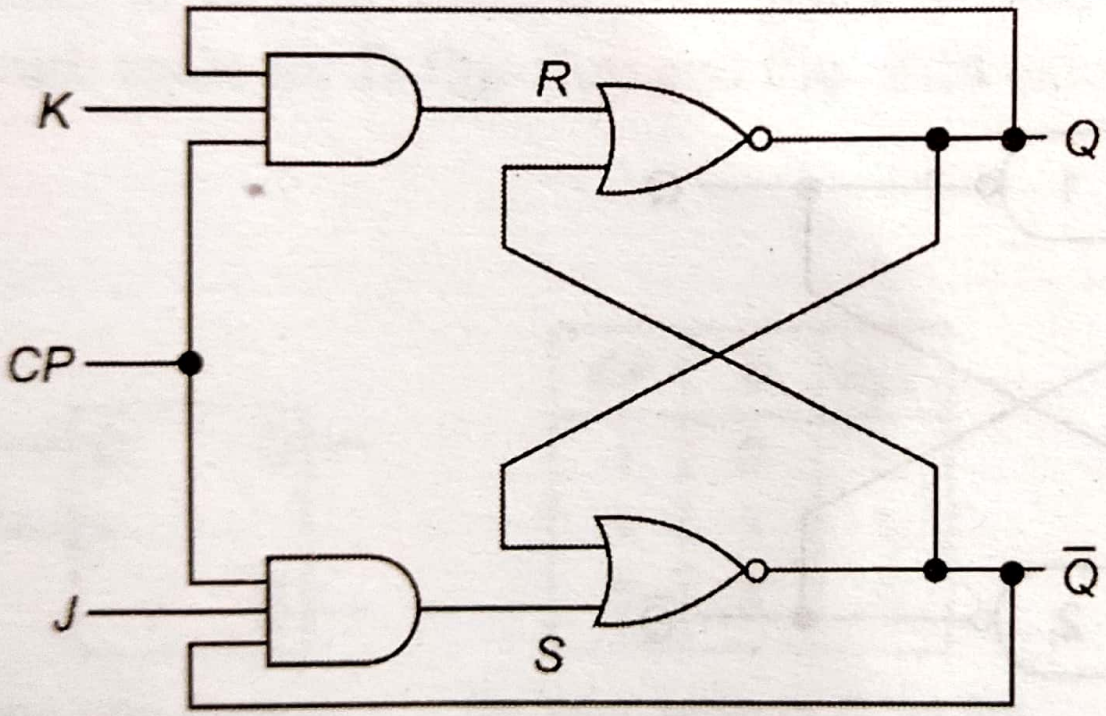



(b) Logic symbol

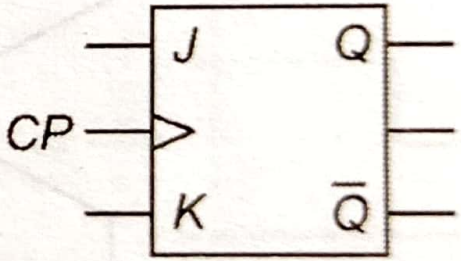
**Fig 9.13**

**Table 9.3:** Truth Table for D latch

| EN | *D* | $Q_n$ | $Q_{n+1}$ | State |
|---|---|---|---|---|
| 1 | 0 | X | 0 | Reset |
| 1 | 1 | X | 1 | Set |
| 0 | X | X | $Q_n$ | No change (NC) |




(a) Clocked JK flip-flop




(b) Logic symbol

| $Q_n$ | J | K | $Q_{n+1}$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

(c) Tru

**Fig. 9.18**

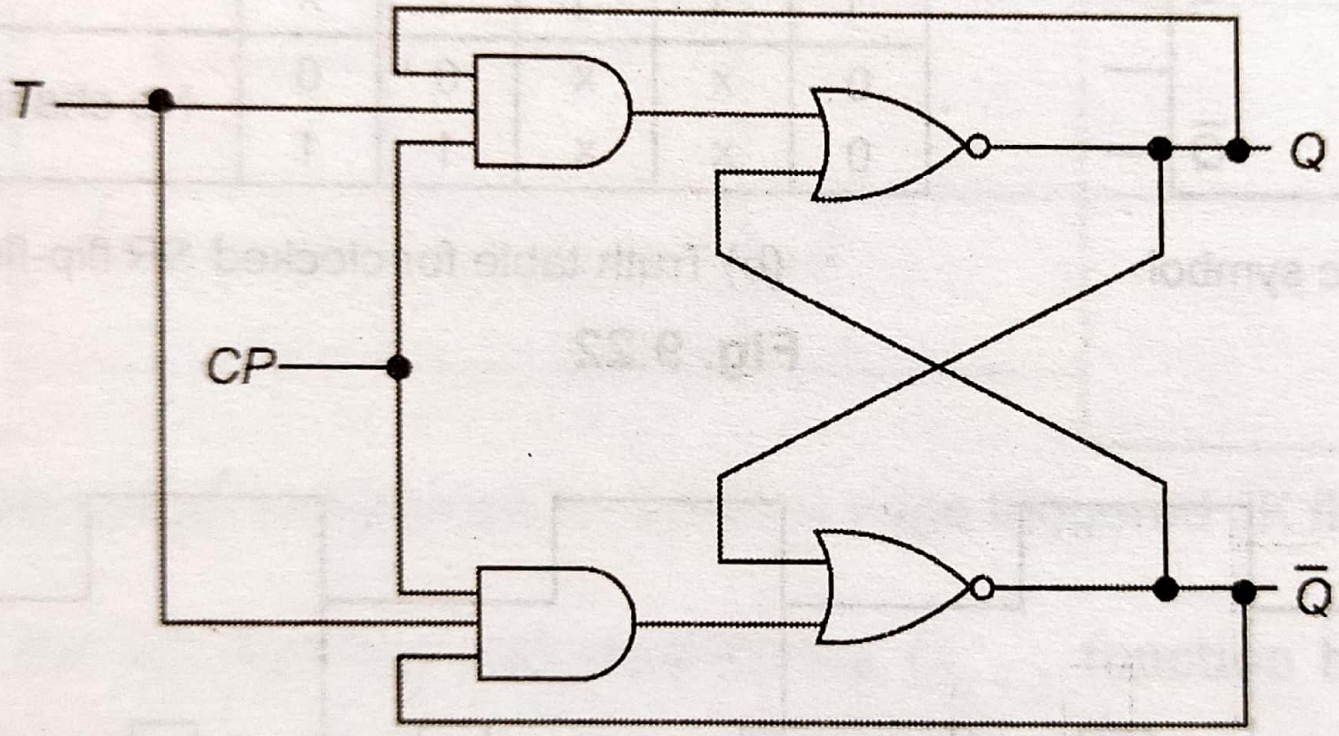
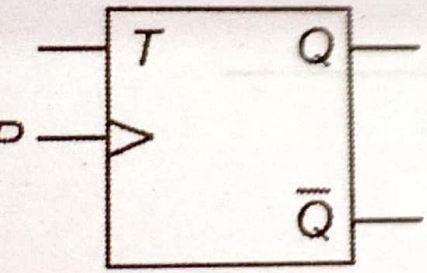


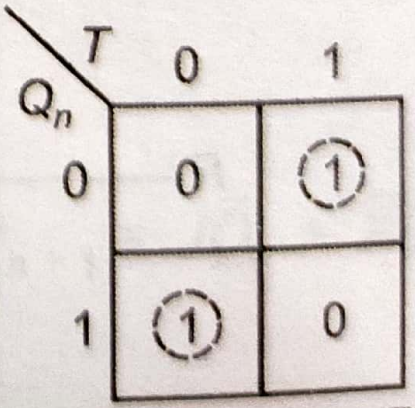

**Fig. 9.25:** Clocked T flip-flop




| $Q_n$ | T | $Q_{n+1}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

≡

| T | $Q_{n+1}$ |
|---|---|
| 0 | $Q_n$ |
| 1 | $\overline{Q_n}$ |



## Master-Slave SR Flip-flop

slave flip-flop is constructed from two flip-flops. One circuit serves as a master and the other and the overall circuit is referred to as a master-slave flip-flop. Fig. 9.27 shows SR master-flop. It consists of a master flip-flop, a slave flip-flop, and an inverter. Both the flip-flops are vel triggered, but inverter connected at the clock input of the slave flip-flop forces it to trigger tive level.

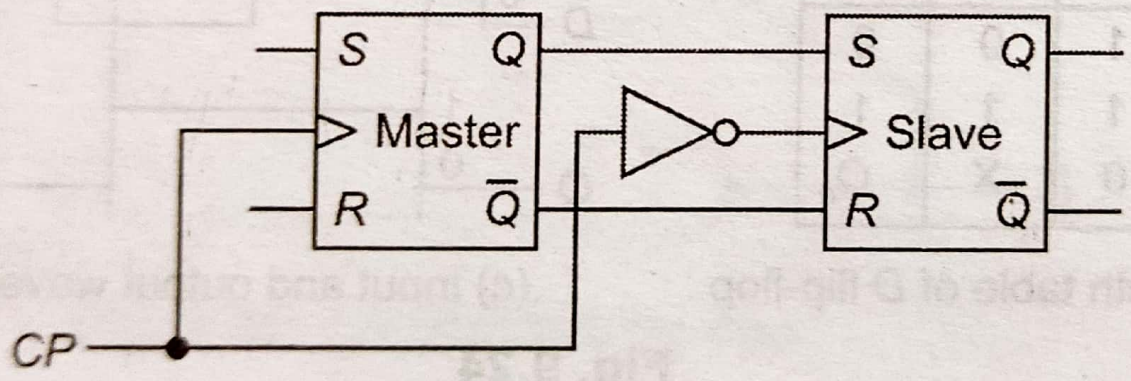



**Fig. 9.27:** Master-slave SR flip-flop

## 9.3.11 Master-Slave JK Flip-flop

The master-slave combination can be constructed for any type of flip-flop. Fig. 9.29 shows one way to build a JK master-slave flip-flop. It consists of clocked JK flip-flop as a master and clocked SR flip-flop as a slave. Like SR master-slave, the output of the master flip-flop is fed as an input to the slave flip-flop. As shown in the Fig. 9.29, clock signal is connected directly to the master flip-flop, but it is connected through inverter to the slave flip-flop. Therefore, the information present at the J and K inputs is transmitted to the output of master flip-flop on the positive clock pulse and it is held there until the negative clock pulse occurs, after which it is allowed to pass through to the output of slave flip-flop. The output of the slave flip-flop is connected as a third input of the master JK flip-flop.

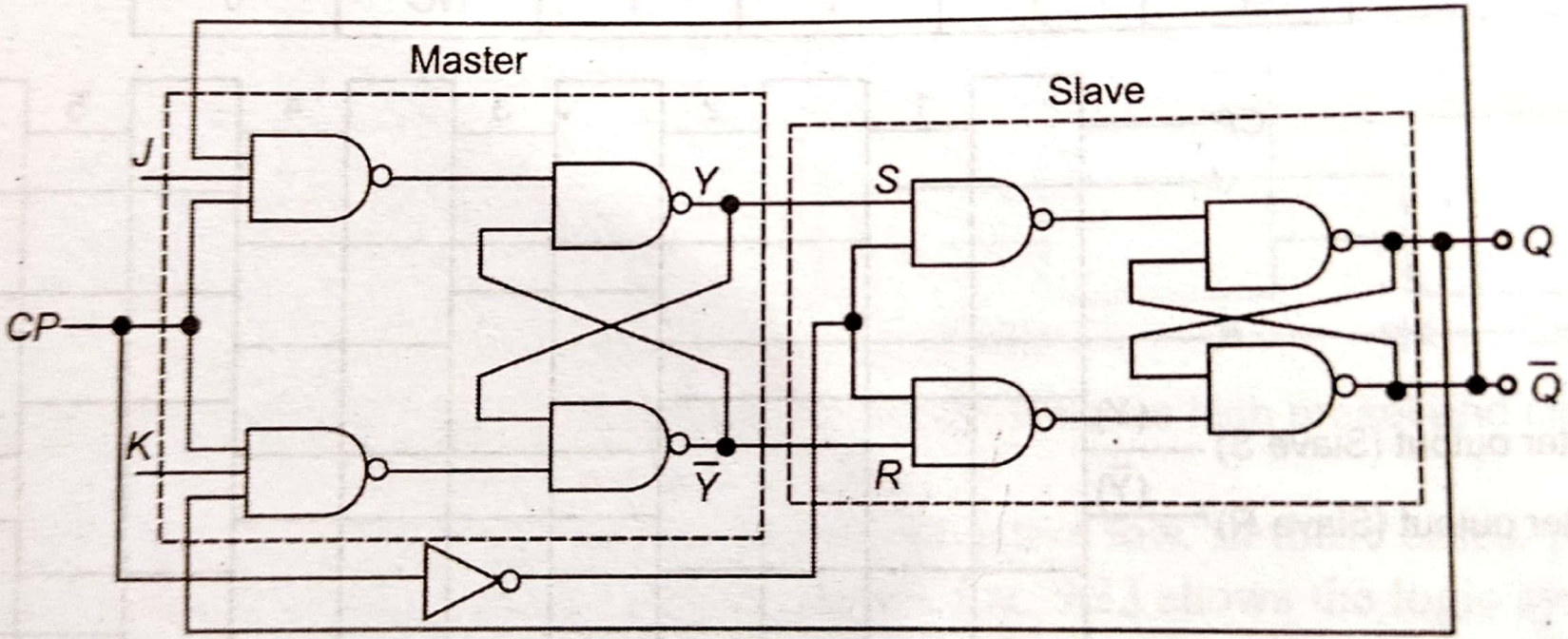



**Fig. 9.29:** Master-slave JK flip-flop

## 9.4 FLIP-FLOP CONVERSIONS

It is possible to convert one flip-flop into another the flip-flop with some additional gates or simply doing some extra connection. Let us see few conversions among flip-flops.

### 9.4.1 Flip-flop to D Flip-flop

The excitation table for above conversion is as shown in Table 9.5 (a).

**Table 9.5 (a)**

| Input | Present state | Next state | Flip-flop inputs | |
|---|---|---|---|---|
| D | $Q_n$ | $Q_{n+1}$ | *S* | *R* |
| 0 | 0 | 0 | 0 | X |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | X | 0 |

**For S**

| D \ $Q_n$ | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | X |

**For *R***

| D \ $Q_n$ | 0 | 1 |
|---|---|---|
| 0 | X | 1 |
| 1 | 0 | 0 |

## 9.4.2 SR Flip-flop to JK Flip-flop

The excitation table for above conversion as shown in Table 9.5 (b).

**Table 9.5 (b)**

| Inputs | | Present state | Next state | Flip-flop inputs | |
|---|---|---|---|---|---|
| J | K | $Q_n$ | $Q_{n+1}$ | S | R |
| 0 | 0 | 0 | 0 | 0 | X |
| 0 | 0 | 1 | 1 | X | 0 |
| 0 | 1 | 0 | 0 | 0 | X |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | X | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 |

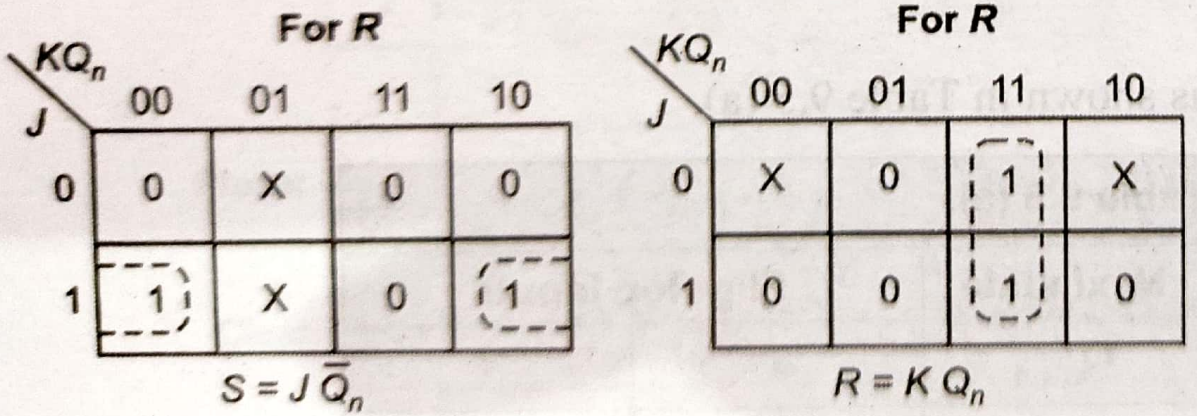



Fig. 9.37: K-map simplification

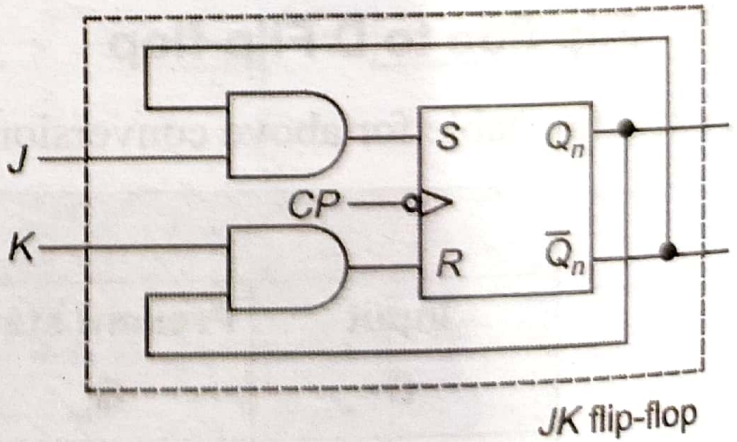



Fig. 9.38: Logic diagram—SR to JK

## 9.4.3 SR Flip-Flop to T Flip-Flop

The excitation table for above conversion is as shown in the Table 9.5 (c)

**Table 9.5(c)**

| Input | Present state | Next state | N Flip-flop inputs | |
|---|---|---|---|---|
| T | $Q_n$ | $Q_{n+1}$ | S | R |
| 0 | 0 | 0 | 0 | X |
| 0 | 1 | 1 | X | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |

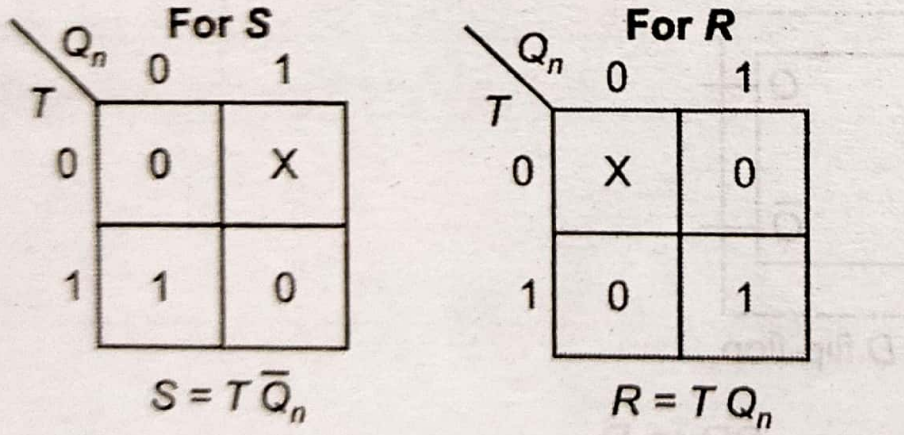



**Fig. 9.39:** K-map Simplification

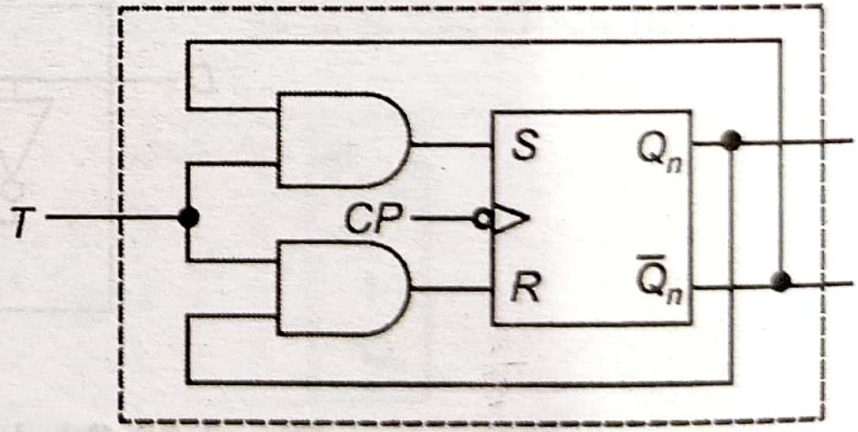



**Fig. 9.40:** Logic Diagram—SR to T

## 9.4.4 JK FLIP-FLOP TO T FLIP-FLOP

The excitation table for above conversion is as shown in Table 9.6(a).

**Table 9.6(a)**

| Input | Present state | Next state | N Flip-flop inputs | |
|---|---|---|---|---|
| T | $Q_n$ | $Q_{n+1}$ | $J_A$ | $K_A$ |
| 0 | 0 | 0 | 0 | X |
| 0 | 1 | 1 | X | 0 |
| 1 | 0 | 1 | 1 | X |
| 1 | 1 | 0 | 0 | 1 |

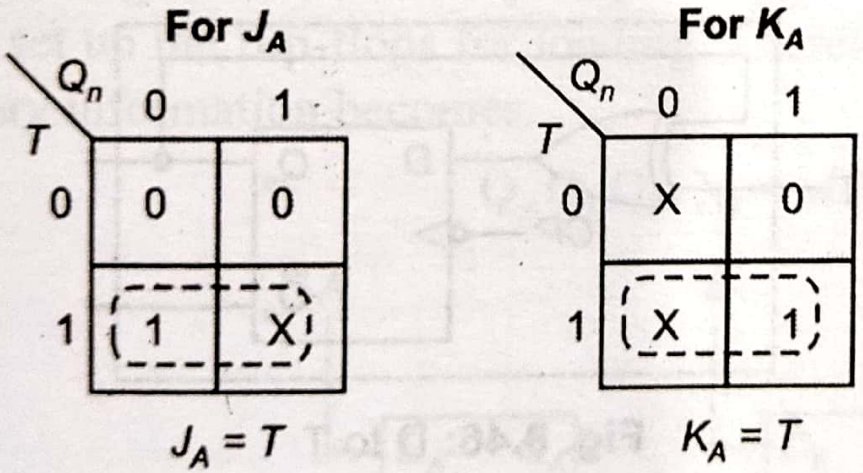



**Fig. 9.41:** K-map Simplification

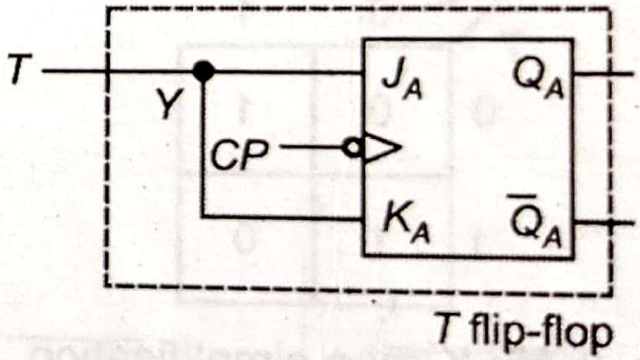



**Fig. 9.42:** Logic Diagram

## 9.4.5 JK Flip-flop to D Flip-flop

The excitation table for above conversion is as shown in the table 9.6(b).

**Table 9.6(b)**

| Input | Present state | Next state | Flip-flop inputs | |
|---|---|---|---|---|
| **D** | $Q_n$ | $Q_{n+1}$ | **J** | **K** |
| 0 | 0 | 0 | 0 | X |
| 0 | 1 | 0 | X | 1 |
| 1 | 0 | 1 | 1 | X |
| 1 | 1 | 1 | X | 0 |

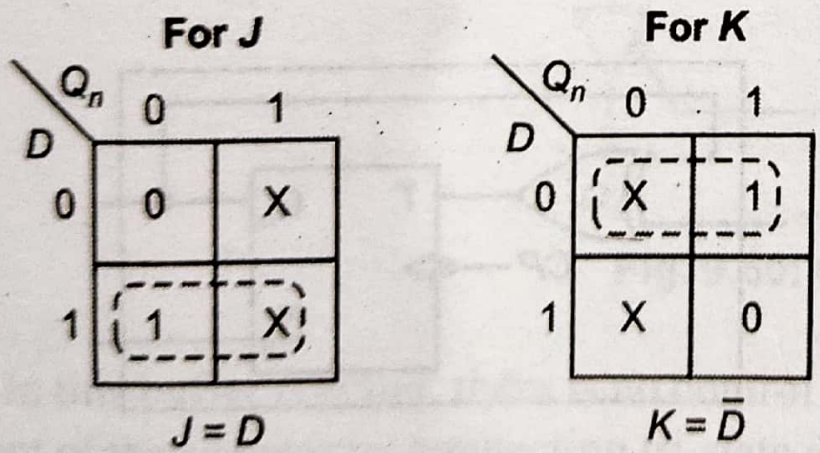



**Fig. 9.43:** K-map Simplification

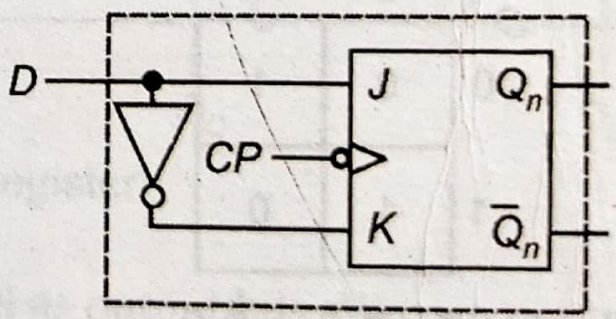



**Fig. .44:** Logic Diagram—JK to D

## 9.4.6 D Flip-Flop to T Flip-Flop

The excitation table for above conversion is as shown in the Table 9.7

**Table 9.7**

| Input | Present state | Next state | Flip-flop inputs |
|---|---|---|---|
| T | $Q_n$ | $Q_{n+1}$ | D |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |

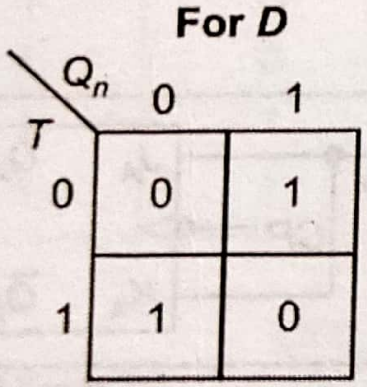



**Fig. 9.45:** K-map simplification

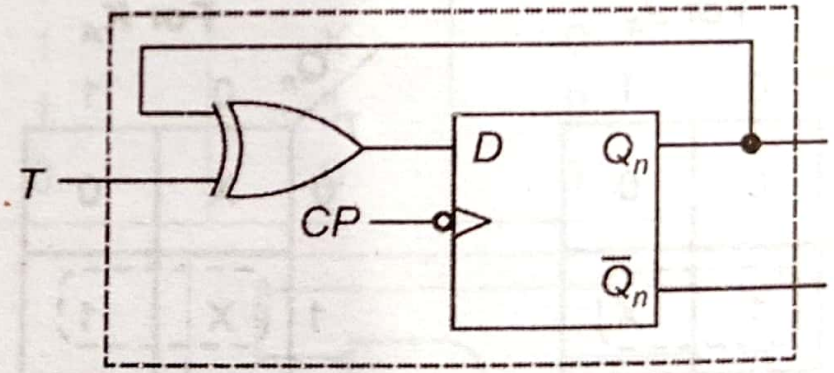



**Fig. 9.46:** D to T

$$D = \overline{T}Q_n + T\overline{Q}_n = T \oplus Q_n$$

## 9.4.7 T Flip-Flop to D Flip-Flop

The excitation table for above conversion is as shown in the Table 9.8

**Table 9.8**

| Input | Present state | Next state | Flip-flop inputs |
|---|---|---|---|
| D | $Q_n$ | $Q_{n+1}$ | T |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |

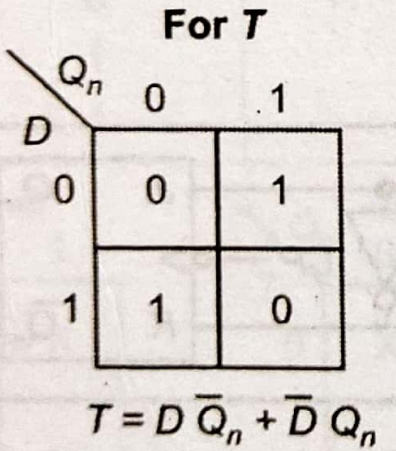



**Fig. 9.47:** K-map Simplification

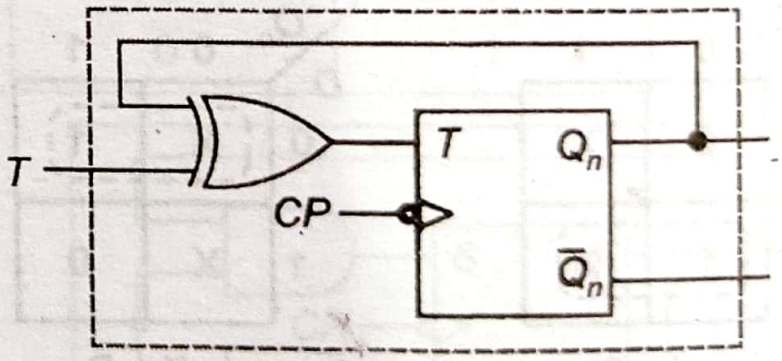


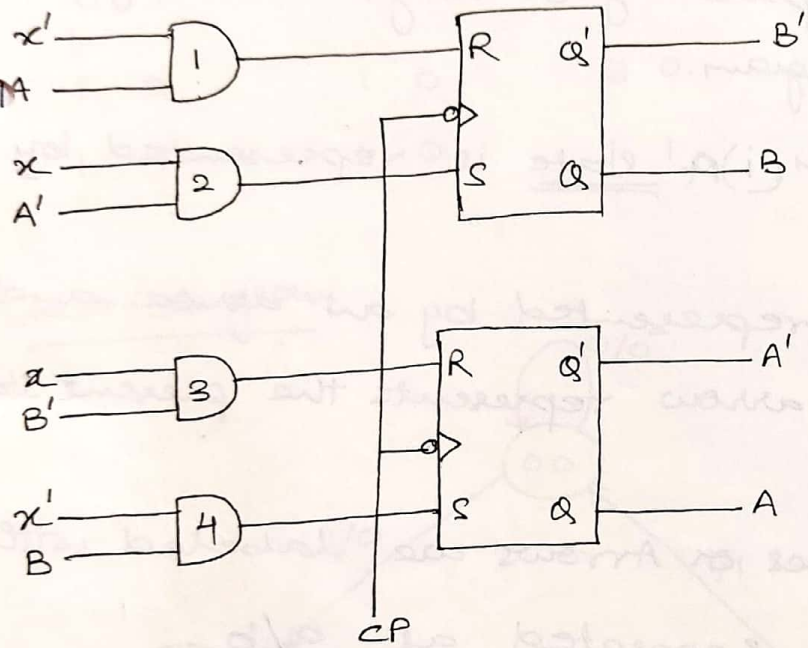

**Fig. 9.48:** Logic Diagram—*T* to *D*

$$T = D\bar{Q}_n + \bar{D}Q_n$$

1. Excitation Tables
2. Conversion of flip-flops
3. Analysis of clocked sequential circuits.

Ex.

(i) Circuit Diagram


- The Behavior of a sequential circuit is determined from the inputs, the outputs and the states of its flip-flops.
- STATE-TABLE (or Transition Table) :-> State table enumerates the time-sequence of inputs, outputs and states of the flip-flops involved in the circuit
- It consists of three sections: present state, next state and output.
- present state :-> It represents or describes the states of flip-flops before the occurance of a clock-pulse.

- The next state represents the states of flip-flops after the application of a clock pulse.
- The output section represents the values of the output variables during the present state.

- STATE-DIAGRAM

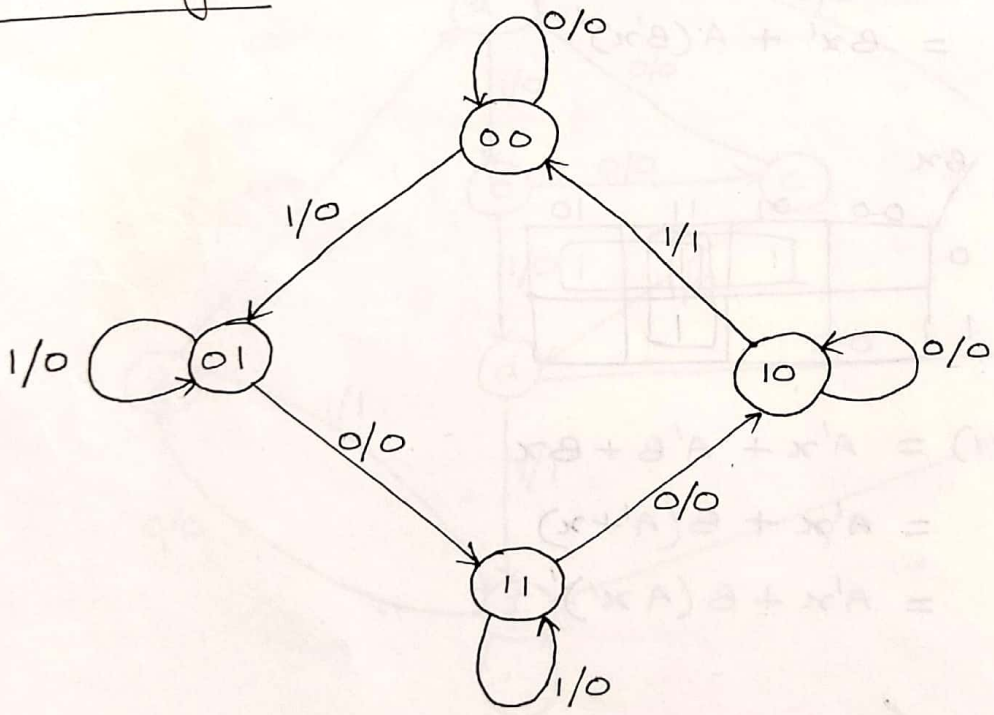
  - The information available in the state-table can be represented graphically or diagramatically with the state-diagram.
  - In a state-diagram (i) A State is represented by a circle.
    (ii) The Transition is represented by an arrow and the direction of arrow represents the present state and next state.
    (iii) The directed lines or Arrows are labeled with two binary numbers separated as a/b.
    where 'a' represents the input value. and
    'b' represents the output value.
    (iv) The arrows starts with 'present state' and end towards the 'next state'.

Ex.

State-Table

| Present State | Next State | | Output | |
|---|---|---|---|---|
| | $x=0$ | $x=1$ | $x=0$ | $x=1$ |
| A B | A B | A B | y | y |
| 0 0 | 0 0 | 0 1 | 0 | 0 |
| 0 1 | 1 1 | 0 1 | 0 | 0 |
| 1 0 | 1 0 | 0 0 | 0 | 1 |
| 1 1 | 1 0 | 1 1 | 0 | 0 |

State-Diagram



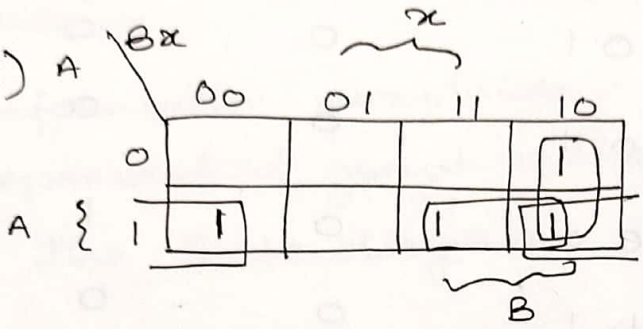

STATE-EQUATIONS

can be properly obtained by solving K-map for $A(t+1)$ and $B(t+1)$ with 3-variable K-map of A, B and x

$$\begin{cases} A(t+1) = (A'B + AB' + AB)x' + ABx \\ B(t+1) = \end{cases}$$

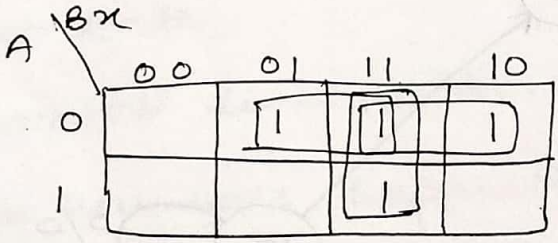
K-Maps

(1.) $A(t+1)$




$$A(t+1) = Ax' + AB + Bx'$$
$$= Bx' + A(B+x')$$
$$= Bx' + A(B'x)'$$

(2.) $B(t+1)$




$$B(t+1) = A'x + A'B + Bx$$
$$= A'x + B(A'+x)$$
$$= A'x + B(Ax')'$$

# State Reduction and State Assignment

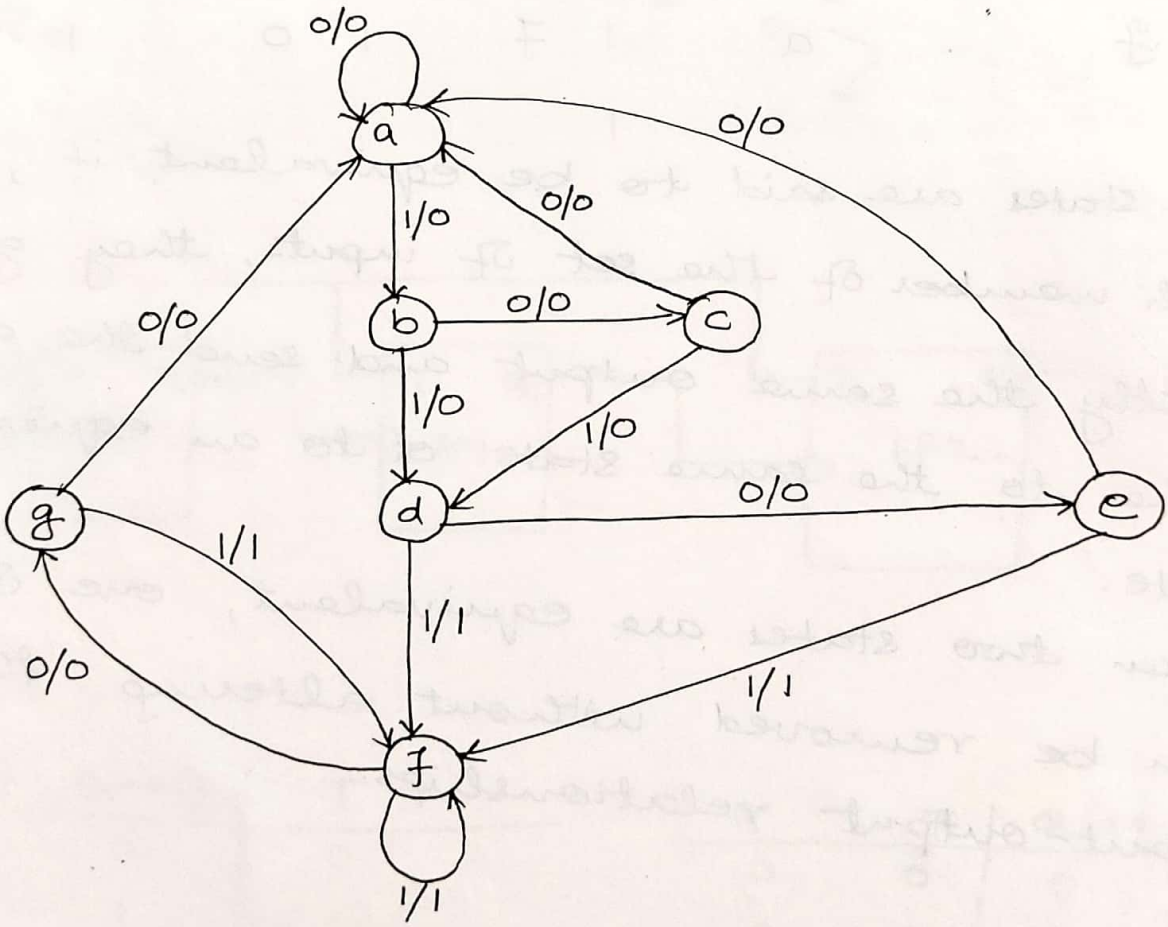
No. of flip-flops used to implement the circuit depends upon the total number of states.

with $n$ flip-flops, we can represent $2^n$ no. of states.

Or we can say that no. of flip-flops reqd. can be obtained by the factor of $2^n$.

i.e. A minimum of $n$ flip-flops are required to represent $\leq 2^n$ states.

Ex.



State Table for the Example.

| Present State | Next State | | Output | |
|---|---|---|---|---|
| | $x=0$ | $x=1$ | $x=0$ | $x=1$ |
| a | a | b | 0 | 0 |
| b | c | d | 0 | 0 |
| c | a | d | 0 | 0 |
| d | e | f | 0 | 1 |
| e | a | f | 0 | 1 |
| f | g | f | 0 | 1 |
| g | a | f | 0 | 1 |

Two states are said to be equivalent if, for each member of the set of inputs, they give exactly the same output and send the circuit either to the same state or to an equivalent state.

When two states are equivalent, one of them can be removed without altering the input-output relationships.

# Synchronous Counters

## Example 1. 3-Bit Binary Up Counter

Steps: → (i) State Diagram
(ii) State Table (PS, i/p and NS) | Flip flop Inputs.
(iii) Solve for Flip Flop i/Ps (using Kmap)
(iv) Implement accd. to the equations.

## Example 2. Random Sequence

3, 2, 4, 5, 7, and Repeat

Sol$^n$ (i) State Table

| Present State | Next State | Flip flop Inputs |
|---|---|---|

*Imp :→ use of Don't care Conditions to solve the K-map.

(ii) Solve for Flip Flop inputs

(iii) Implement.

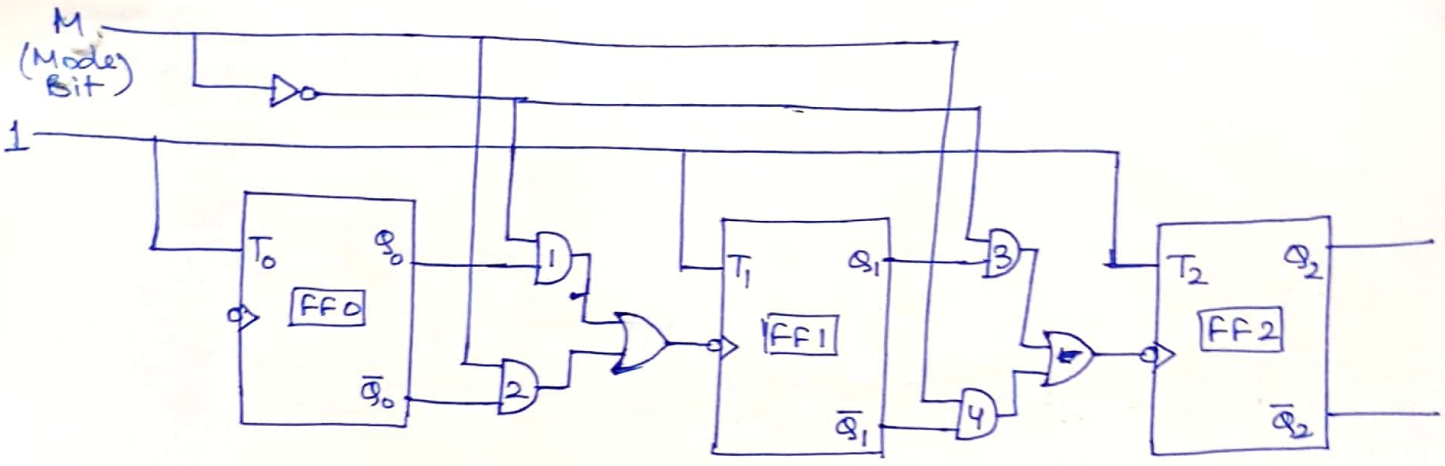
## Example 3. Modulus-Counter
## Decade Counter

Sol$^n$ (i) State Table → PS | NS | FF Inputs

*Imp :→ Use of Don't Care Conditions to solve the K-map

(ii) Solve for Flip Flop inputs

(iii) Implement.



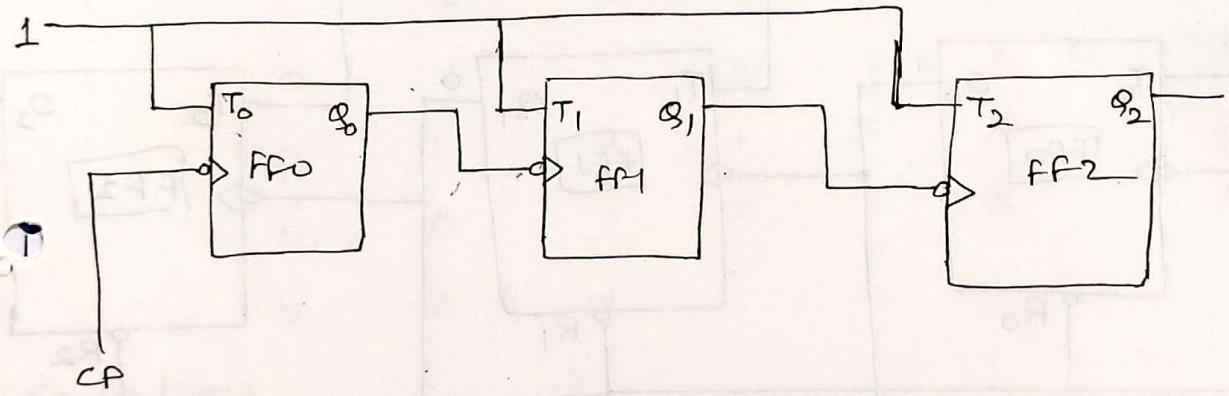

## 3-Bit Up / Down Asynchronous Counter

| Mode Control Bit M | Flip flop Outputs $Q_2$ | $Q_1$ | $Q_0$ | Decimal equivalent |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 1 | 1 | 3 |
| 0 | 1 | 0 | 0 | 4 |
| 0 | 1 | 0 | 1 | 5 |
| 0 | 1 | 1 | 0 | 6 |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 7 |
| 1 | 1 | 1 | 0 | 6 |
| 1 | 1 | 1 | 1 | 5 |
| 1 | 1 | 0 | 0 | 4 |
| 1 | 1 | 0 | 1 | 3 |
| 1 | 0 | 1 | 0 | 2 |
| 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 |

# Asynchronous or Ripple Counters

## (i) 3-bit Binary counter

| Counter State | count $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 2 | 0 | 1 | 0 |
| 3 | 0 | 1 | 1 |
| 4 | 1 | 0 | 0 |
| 5 | 1 | 0 | 1 |
| 6 | 1 | 1 | 0 |
| 7 | 1 | 1 | 1 |



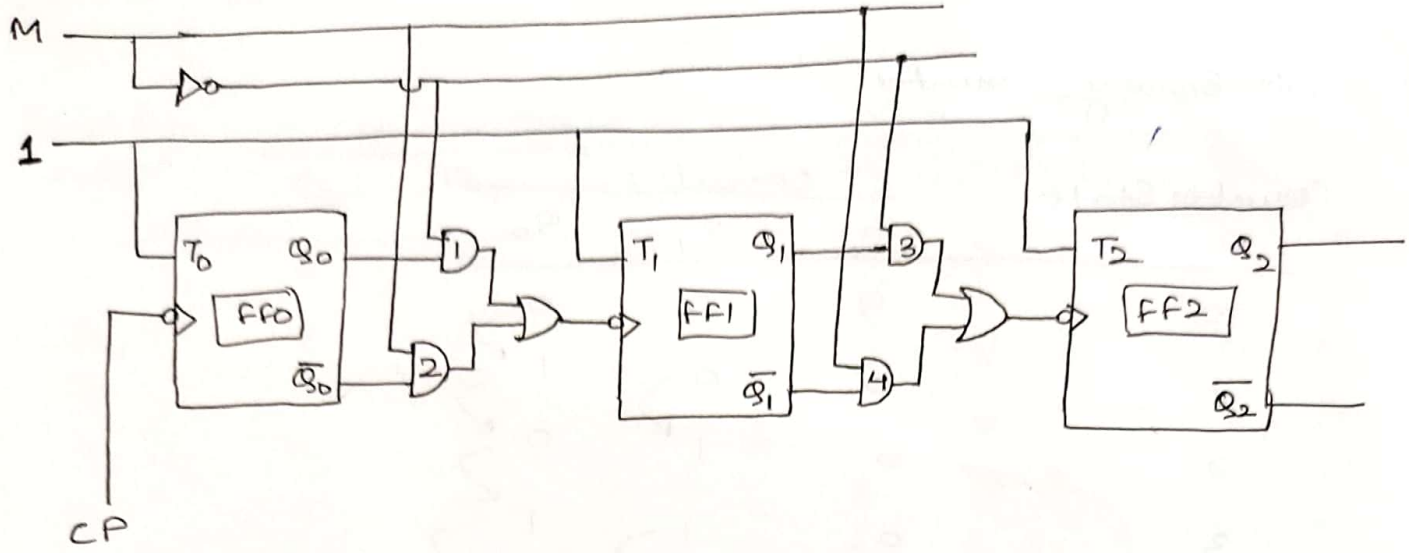

| $Q(t)$ | $Q(t+1)$ | T |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

⇒

| Q | T | Q(t+1) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

* Home Assignment :→ what if we change the clock pulse to be +ve edge triggered rather than -ve edge triggering.

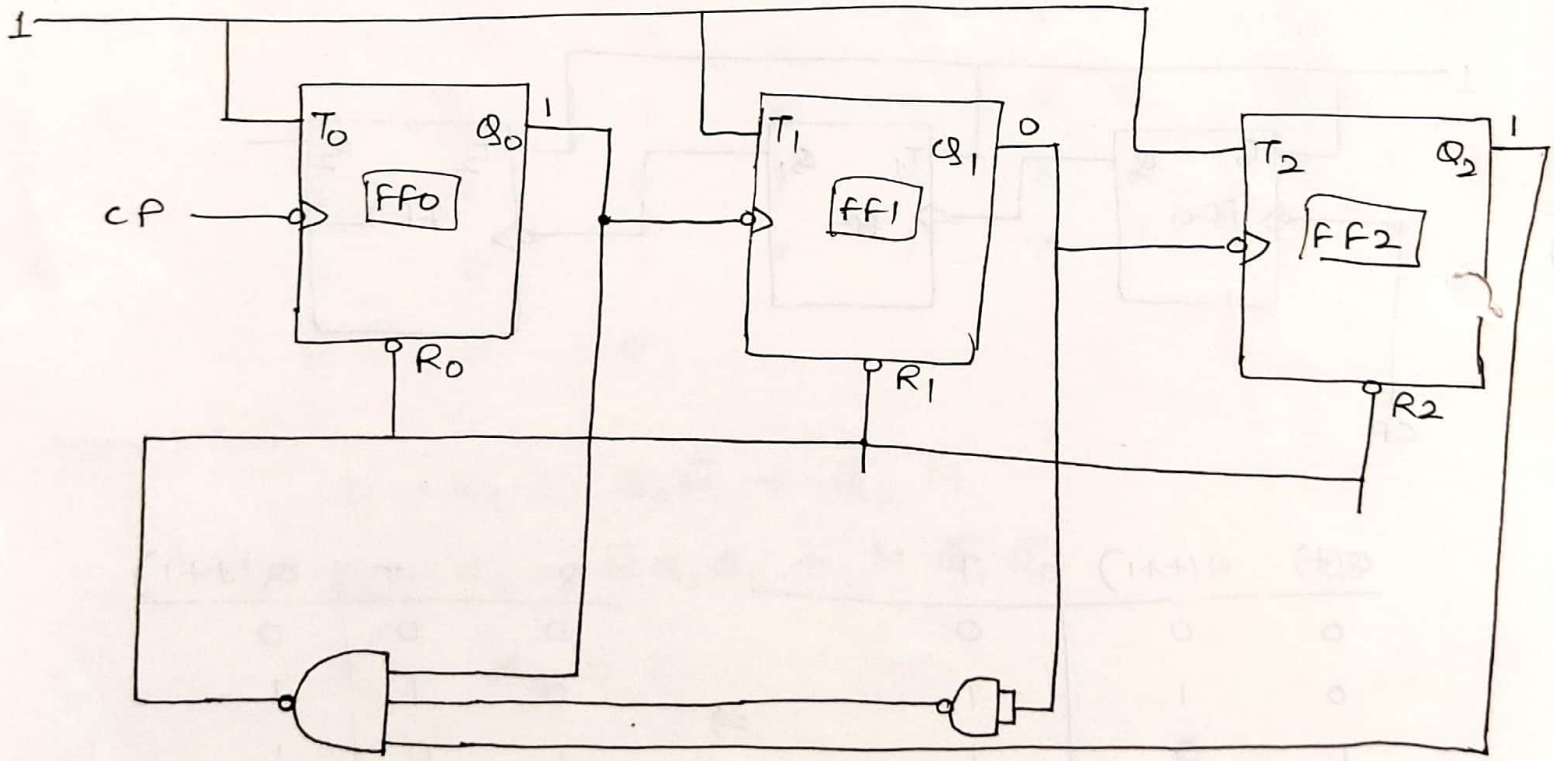
## Asynchronous 3-bit Binary up/down counter.




---

- Mod-6 counter – Asynchronous

  - Count from 000 to 101
  - After 101', counter resets to 000



# 3-bit Synchronous Binary up/down Counter with direction control M

| Direction Control M | Counter State | | | Flip Flop Inputs | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $Q_2$ | $Q_1$ | $Q_0$ | $J_0$ | $K_0$ | $J_1$ | $K_1$ | $J_2$ | $K_2$ |
| 0 | 0 | 0 | 0 | 1 | X | 0 | X | 0 | X |
| 0 | 0 | 0 | 1 | 0 | X | 1 | X | 0 | X |
| 0 | 0 | 1 | 0 | 1 | X | X | 0 | 0 | X |
| 0 | 0 | 1 | 1 | X | 1 | X | 1 | 1 | X |
| 0 | 1 | 0 | 0 | 1 | X | 0 | X | X | 0 |
| 0 | 1 | 0 | 1 | X | 1 | 1 | X | X | 0 |
| 0 | 1 | 1 | 0 | 1 | X | X | 0 | X | 0 |
| 0 | 1 | 1 | 1 | X | 1 | 1 | X | X | 0 |
| 1 | 0 | 0 | 0 | 1 | X | 1 | X | 1 | X |
| 1 | 1 | 1 | 1 | X | 1 | X | 0 | X | 0 |
| 1 | 1 | 1 | 0 | 1 | X | X | 1 | X | 0 |
| 1 | 1 | 0 | 1 | X | 1 | 0 | X | X | 0 |
| 1 | 1 | 0 | 0 | 1 | X | 1 | X | X | 1 |
| 1 | 0 | 1 | 1 | X | 1 | X | 0 | 0 | X |
| 1 | 0 | 1 | 0 | 1 | X | X | 1 | 0 | X |
| 1 | 0 | 0 | 1 | X | 1 | X | 1 | 0 | X |
| | 0 | 0 | 0 | | | | | | |

## Equations

$$J_1 = K_1 = Q_0\bar{M} + \bar{Q_0}M$$

$$J_2 = K_2 = \bar{M}Q_1Q_0 + M\bar{Q_1}\bar{Q_0}$$

$$J_0 = K_0 = 1$$

| $Q_2$ | $Q_1$ | $Q_0$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

$Q_0$ → State change (Toggle) with every clock

$Q_1$ → Toggle when $Q_0$ goes from 1 to 0

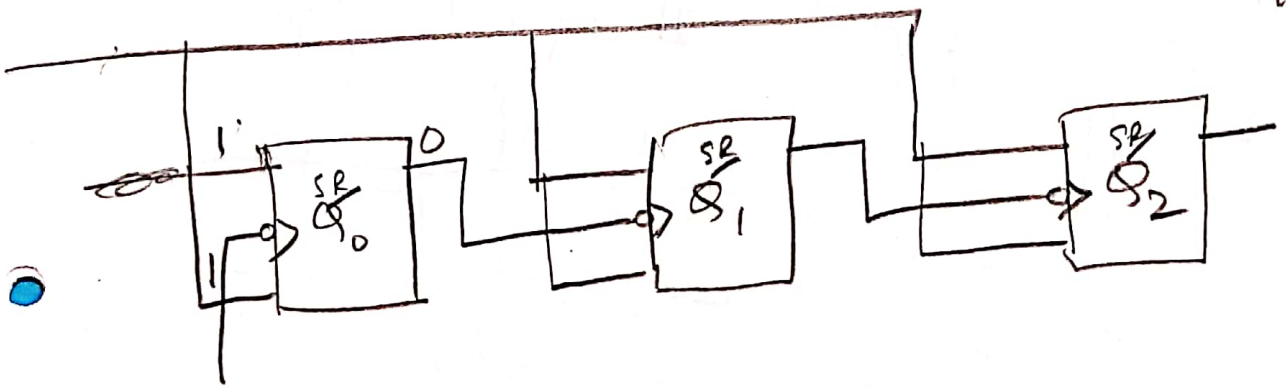
$Q_2$ → Toggle when $Q_1$ goes from 1 to 0

## Binary up/down counter

↳ divide-by-N counter

↓

no. of ffs




| Q | J | K | Q(t+1) | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Same |
| 0 | 0 | 1 | 0 | |
| 0 | 1 | 0 | 1 | Toggle |
| 0 | 1 | 1 | 1 | Same |
| 1 | 0 | 0 | 1 | |
| 1 | 0 | 1 | 0 | |
| 1 | 1 | 0 | 1 | |
| 1 | 1 | 1 | 0 | Toggle |

**Down Counter**

1 → 0 clock Transition

**Up Counter**

0 → 1 clock Transition

# Synchronous Counter

| | PS | | | NS | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | $Q_2$ | $Q_1$ | $Q_0$ | $Q_2$ | $Q_1$ | $Q_0$ | $J_2$ | $K_2$ | $J_1$ | $K_1$ | $J_0$ | $K_0$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | | | | | | |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | | | | | | |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | | | | | | |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | | | | | | |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | | | | | | |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | | | | | | |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | | | | | | |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | | | | | | |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | | | | | | |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | | | | | | |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | | | | | | |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | | | | | | |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | | | | | | |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | | | | | | |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | | | | | | |
| | 0 | 0 | 0 | | | | | | | | | |

↕ UP counter
↕ DOWN counter

using Excitation table of JK-ff

6 K-maps of 4-variables

solve equations for $J_0, K_0, J_1, K_1$ $J_2$ and $K_2$ using 4-i/p variables namely M, $Q_2$, $Q_1$ and $Q_0$.

# Modulus / Mod- Counter

Mod- N Counter
→ counts from 0 to (N-1) in decimal

(1.) for Asynchronous / Counter
ripple
Design reset-logic for 1st invalid-
State.

(2.) for Synchronous counter
Simply follow the PS → NS Table
design steps and consider all
non-valid states as don't care for
solving the k- map.

## BCD / Ripple Counter
Decade

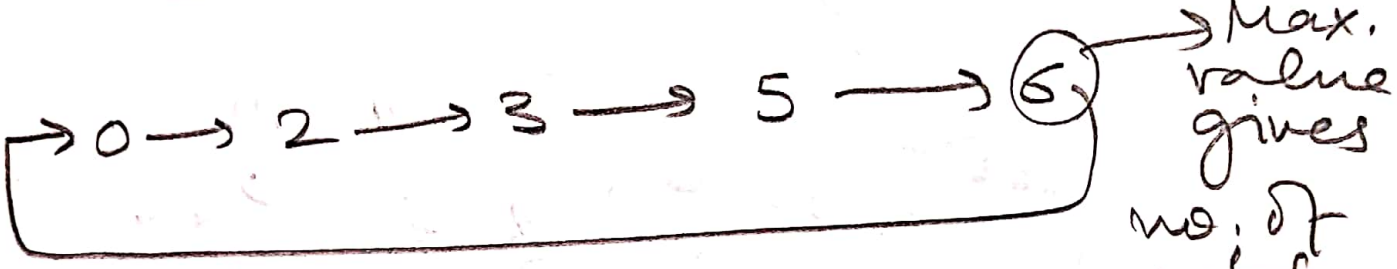
0 → 10 in decimal

0000 → 1010 Valid States

1011 → first invalid State.

Reset logic

## Random Sequence Counter



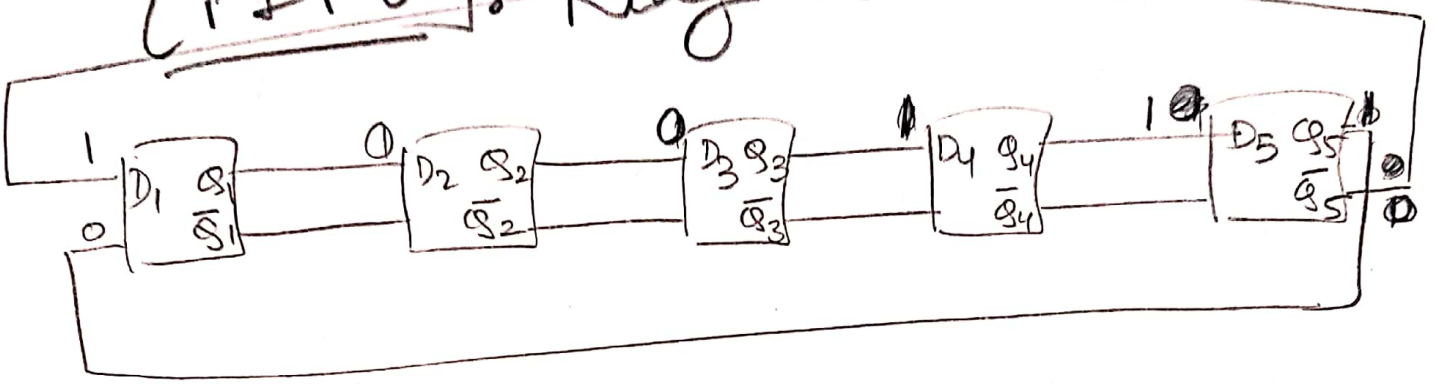

| PS | | | NS | | | Excitations | | | ffs |
|---|---|---|---|---|---|---|---|---|---|
| $Q_2$ | $Q_1$ | $Q_0$ | $Q_2$ | $Q_1$ | $Q_0$ | $T_2$ | $T_1$ | $T_0$ | |
| 0 | 0 | 0 | 0 | 1 | 0 | | | | |
| 0 | 1 | 0 | 0 | 1 | 1 | | | | |
| 0 | 1 | 1 | 1 | 0 | 1 | | | | |
| 1 | 0 | 1 | 1 | 1 | 0 | | | | |
| 1 | 1 | 0 | 0 | 0 | 0 | | | | |

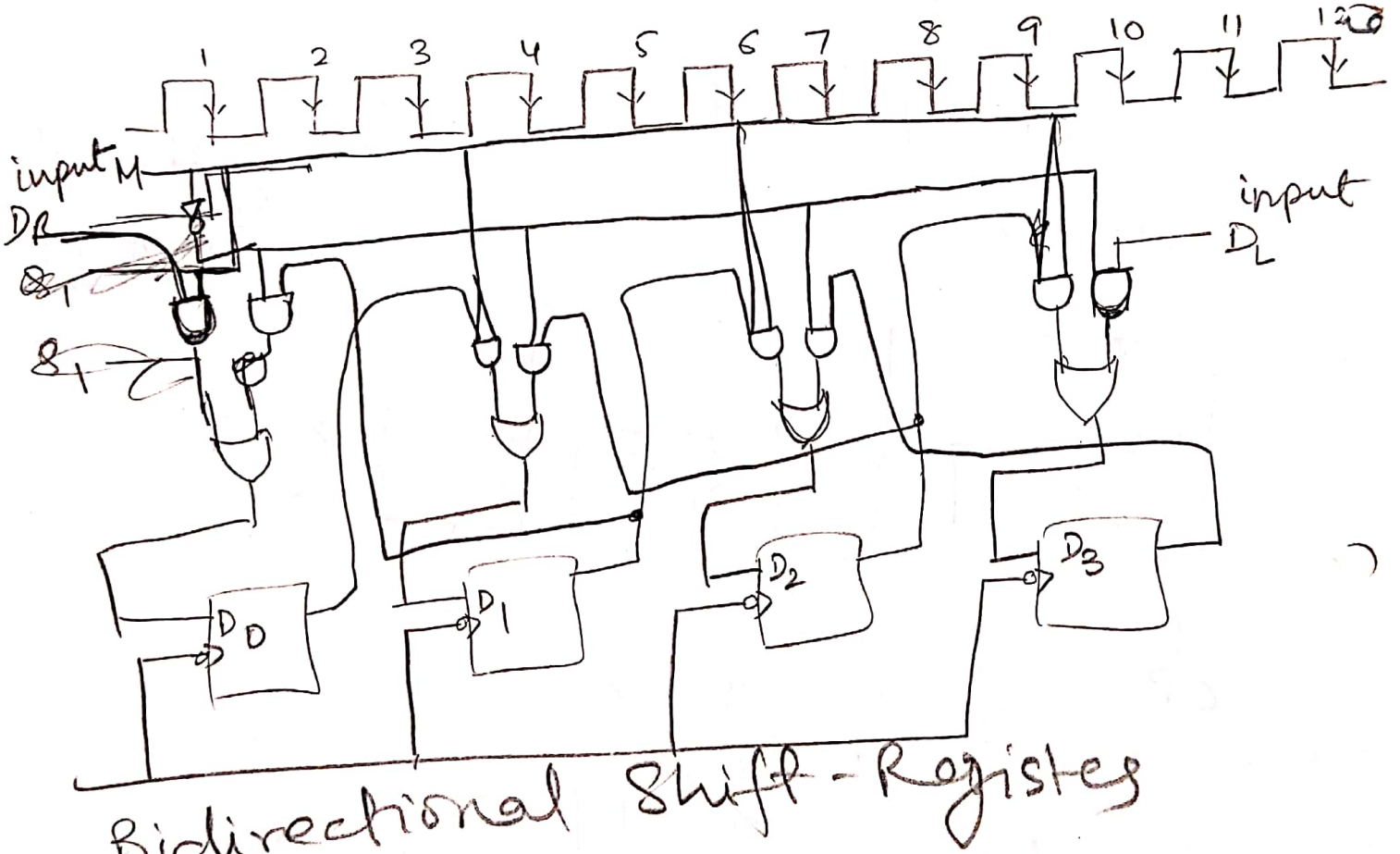
Solve for $T_2$, $T_1$ & $T_0$ and keeping ①, ④ and ⑦ states as don't care conditions for solving k-map.

# Shift Registers

- SIPO
- PISO
- SISO
- PIPO

- Bidirectional Shift Register
- Johnson Counter
- Ring Counter



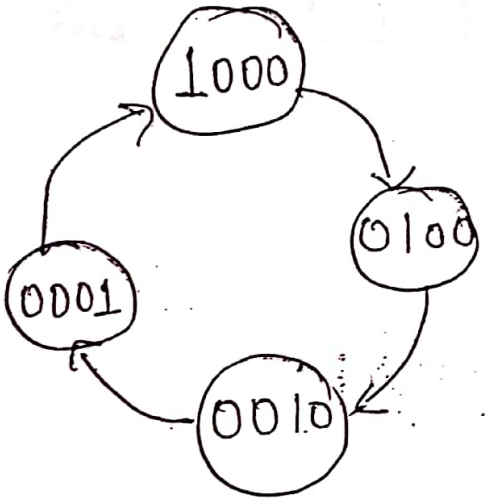

Bidirectional Shift-Registers

**Ring Counter** :→ • Counts no. of pulses.
• divide-by-N Counter
↳ no. of ff's.

**Johnson Counter** → Divide-by-2N Counter
(Twisted Ring)
Moebius Counter

# RING COUNTER :

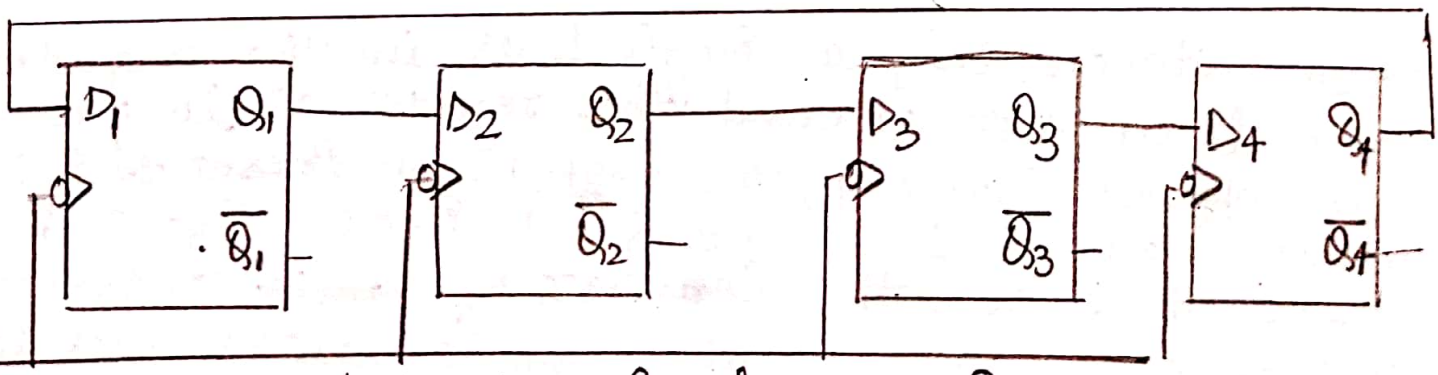
A array of FF's are arranged in a ring and therefore named as Ring Counter.



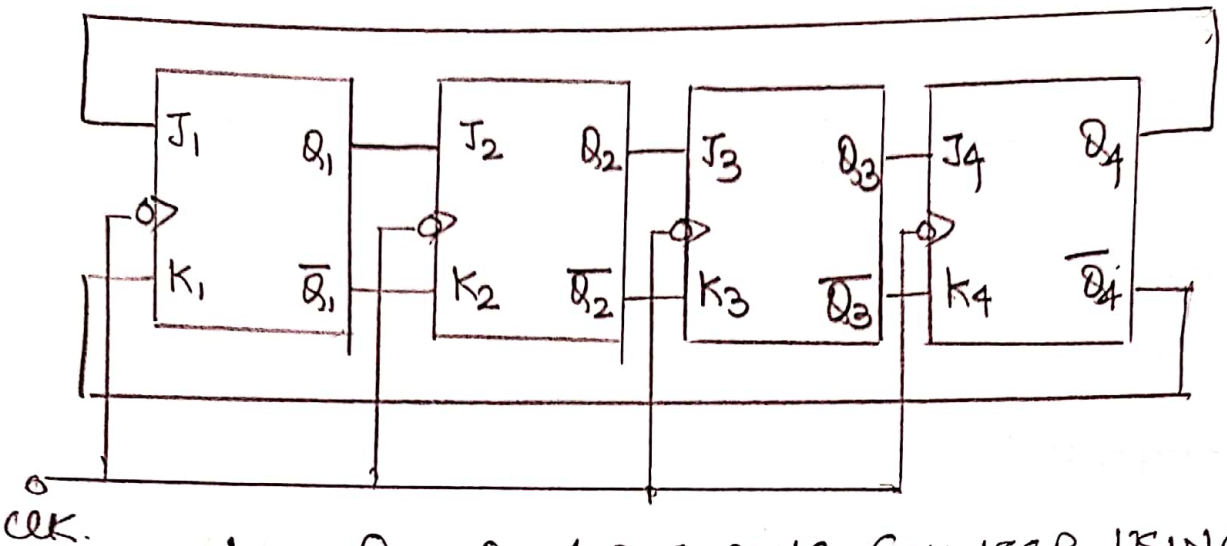

(a) State Diagram

| $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | Clk Pulse |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 0 | 1 | 3. |
| 1 | 0 | 0 | 0 | 4 |
| 0 | 1 | 0 | 0 | 5 |
| 0 | 0 | 1 | 0 | 6 |
| 0 | 0 | 0 | 1 | 7. |

(b) Sequence Table.

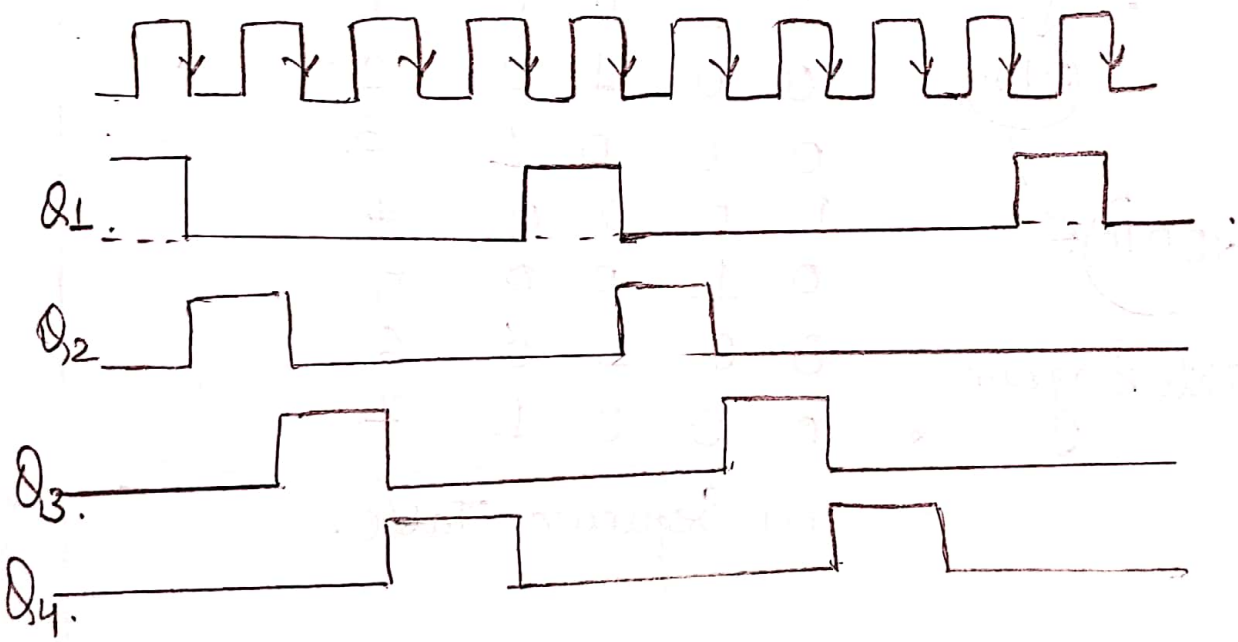



4 Bit Ring Counter using D F.F's.




## Logic Diag. Of 4 Bit Ring Counter Using JK.

Mod. of Ring Counter = No. of FF's used in Counter
An n-bit Ring Counter can Count ~~only~~ n bits.
n bit Ripple Counter can Count only $2^n$ bits.




In most instances, only a Single 1 is in the register & is made to Circulate Around the register as long as clk pulses are applied. Initially, the first FF is Preset to a 1 So initial State is 1000 i.e $Q_1 = 1$, $Q_2 = 0$, $Q_3 = 0$ & $Q_4 = 0$.
After each clk pulse, the Contents of ~~shift~~ register are Shifted to right by one bit & $Q_4$ is Shifted back to $Q_1$.
Sequence Repeats after 4 clk Pulses.
No Decoder is required.

# TWISTED RING COUNTER (JOHNSON COUNTER).

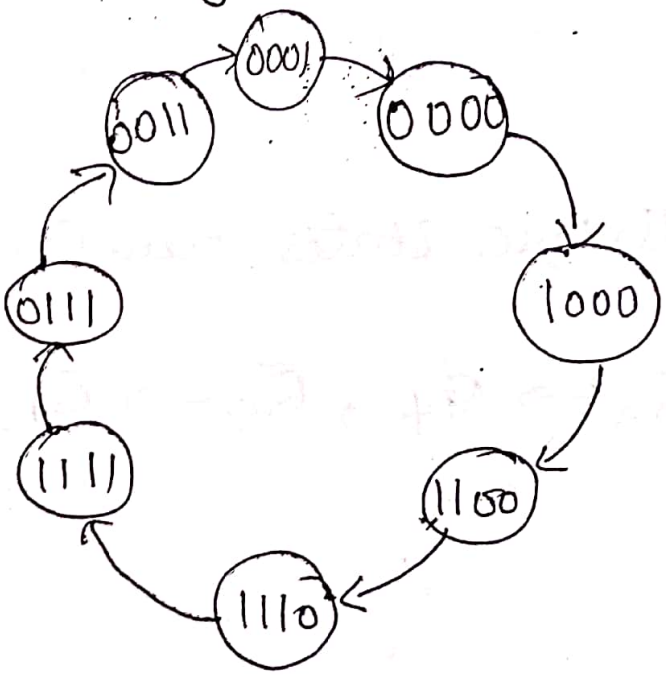
Counter is obtained from Serial-in, Serial out Shift register by providing f/b from the inverted O/P of last FF to D i/p of first FF.

Q o/p $\xrightarrow{\text{Connected}}$ D i/p of Next Stage.

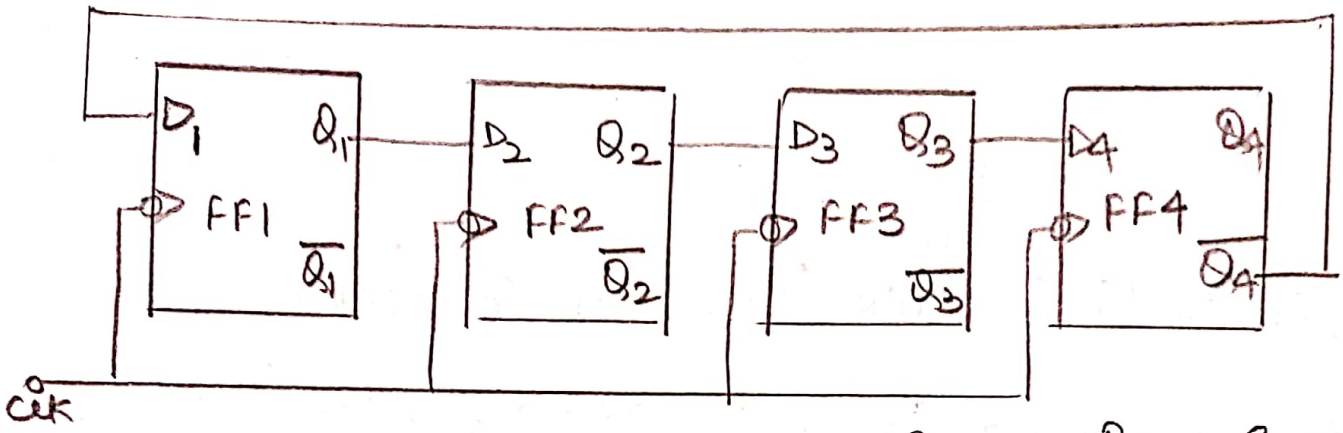
But $\overline{Q}$ o/p of Last stage $\xrightarrow{\text{Connected}}$ D o/p of first stage

Therefore Named as twisted Ring Counter.

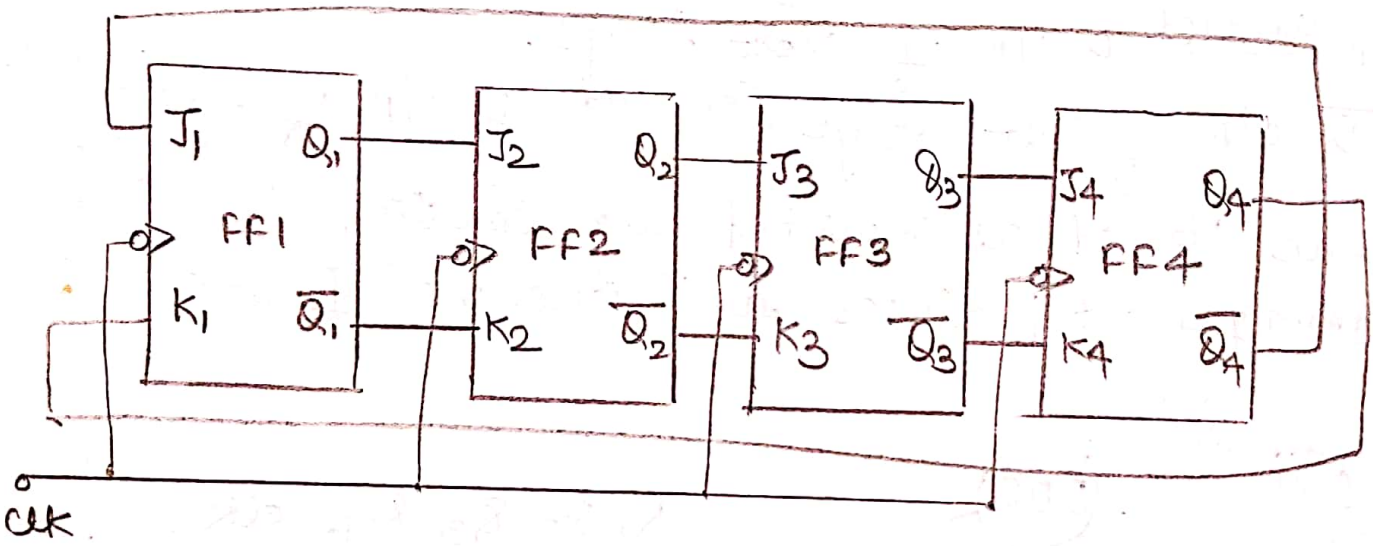
f/b arrangement produces a Unique no. of states.




| $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | clk. |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 2 |
| 1 | 1 | 1 | 0 | 3 |
| 1 | 1 | 1 | 1 | 4 |
| 0 | 1 | 1 | 1 | 5 |
| 0 | 0 | 1 | 1 | 6 |
| 0 | 0 | 0 | 1 | 7 |
| 0 | 0 | 0 | 0 | 8 |
| 1 | 0 | 0 | 0 | 9. |



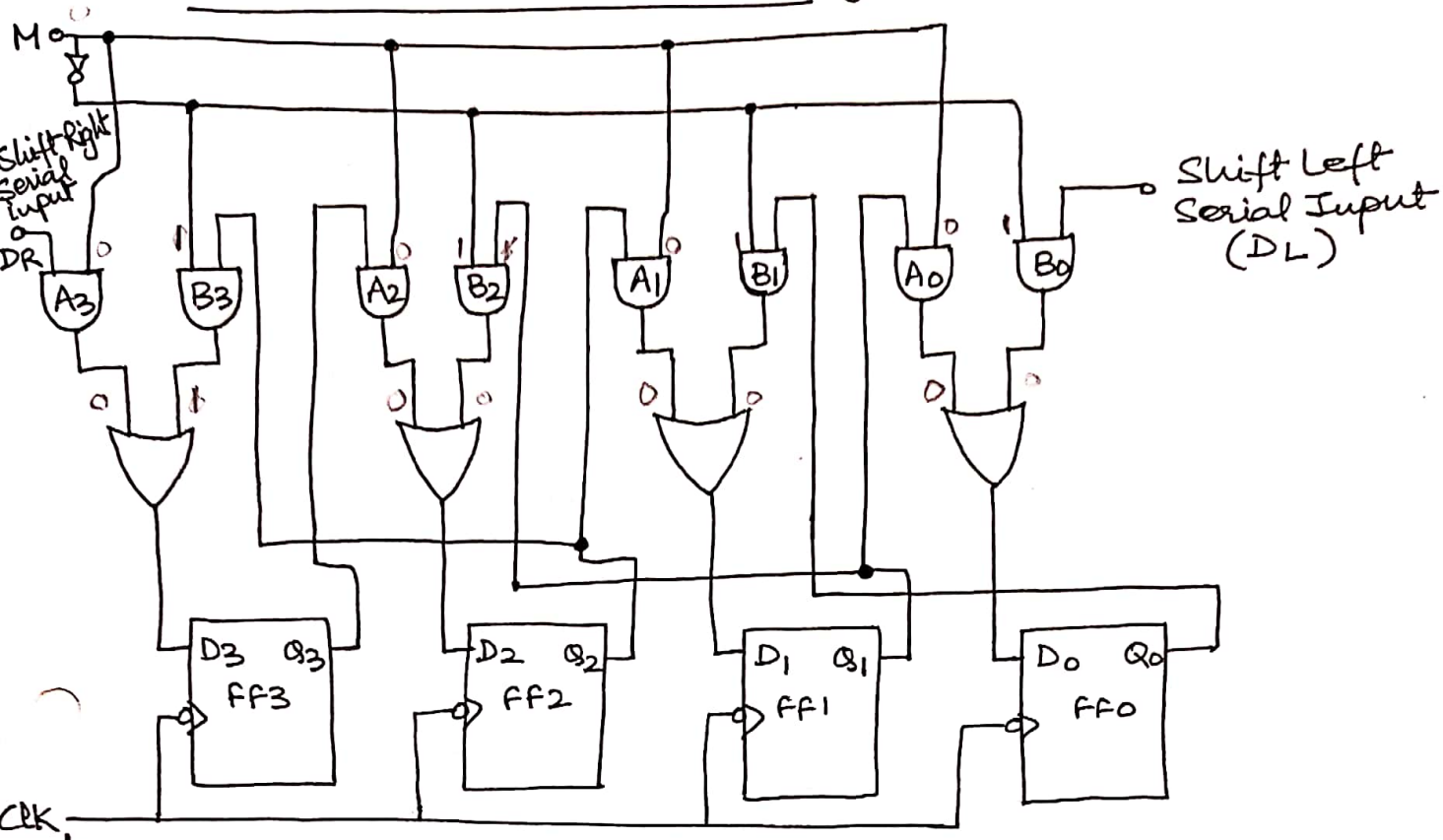

LOGIC DIAG. OF 4 BIT TWISTED RING COUNTER




N FF Johnson ⟶ 2n Unique states can Count ~~upto~~ 2n pulses.

$Q_1 \longrightarrow Q_2$ ; $Q_2 \longrightarrow Q_3$ , $Q_3 \longrightarrow Q_4$ , $\overline{Q_4} \longrightarrow Q_1$.

# 4-Bit Bidirectional Shift-Register




Mode control bit

M = 1 Right Shift

M = 0 Left Shift